WHAT IS CLAIMED:

I	1. 1	A method of translating a plurality of basic blocks of program code,
2	comprising:	
3	C	decoding a plurality of basic blocks of program code;
4	<u> </u>	generating an intermediate representation for each of said basic blocks of
5	program	code;
6	ŗ	performing a partial dead code elimination optimization on said
7	intermed	diate representation to generate an optimized intermediate representation;
8	8	generating target code from said optimized intermediate representation.
1	2.	The method of claim 1, wherein the partial dead code elimination
2	optimization is	performed on basic blocks ending in non-computed branches or computed
3	jumps.	
1	3.	The method of claim 2, wherein the partial dead code elimination
2	optimization co	mprises:
3	i	dentifying partially dead register definitions within a basic block;
4	a	applying a recursive marking algorithm to mark the child nodes of
5	identifie	ed partially dead register definitions to produce a set of partially dead
6	nodes; a	nd .
7	I	performing code motion optimization algorithm to generate an optimized
8	intermed	diate representation providing an optimized order for generating target
9	code.	

1	4.	The method of claim 3, wherein the partially dead register definition
2	identifying st	tep comprises:
3		for a register definition in a basic block, performing liveness analysis of
4	said r	egister definition in successor basic blocks containing said non-computed
5	branc	hes or computed jumps; and
6	•	identifying said register definition as being partially dead if said register
7	defin	ition is dead in at least one successor basic block and live in at least one
8	other	successor basic block.
1	5.	The method of claim 4, further comprising forming a set of identified
2	partially dead	d register definitions.
1	6.	The method of claim 5, further comprising applying a recursive marking
2	algorithm to	identify partially dead child nodes in the intermediate representation of
3	identified par	rtially dead register definitions.
1	7.	The method of claim 6, wherein said recursive marking algorithm
2	identifies a n	ode in the intermediate representation as a partially dead child node by
3	ensuring that	the node is not referenced by either a live node or a live register definition
1	8.	The method of claim 6, wherein said recursive marking algorithm
2	identifies par	tially dead child nodes as those nodes which are only referenced in the
3	intermediate	representation by other partially dead nodes or partially dead register
4	definitions.	

1	9.	The method of claim 8, wherein said recursive marking algorithm includes
2	the steps of:	
3		determining a dead count for a child node, wherein the dead count is the
4	numb	er of partially dead nodes referencing the child node in the intermediate
5	repres	sentation;
6		determining a reference count for the child node, wherein the reference
7	count	is the number of references to the child node in the intermediate
8 -	repres	sentation; and
9	•	identifying a child node as a partially dead when its dead count equals its
10	refere	nce count.
1	10.	The method of claim 6, wherein said recursive marking algorithm further
2	recursively id	lentifies whether the child nodes of identified partially dead child nodes are
3	also partially	dead.
1	11.	The method of claim 3, the code motion optimization algorithm
2	comprises:	
3		for each identified partially dead register definition,
4		determining a path of nodes in the intermediate representation for said
5	partia	lly dead register definition which are live,
6		discarding the nodes in the intermediate representation for said partially
7	dead	register definition which are dead, and
8		determining partially live paths of nodes in the intermediate representation
9	for sa	id partially dead register definition and moving corresponding nodes into
10	said p	partially live paths, wherein the nodes in the partially live paths are partially

11 -	dead nodes, further wherein a partially live path of nodes exists for each	
12	respective branch or jump.	
1	12. The method of claim 11, wherein each node in the intermediate	
2	representation includes an associated variable which identifies with which partially live	
3	path of nodes it is associated.	
1	13. The method of claim 11, wherein said target code generating step	
2	comprises:	
3	initially generating target code for all fully live nodes of a partially dead	
4	register definition; and	
5	next generating target code for said partially live paths of nodes in the	
6	intermediate representation for said partially dead register definition.	
1	14. The method of claim 11, wherein the code motion optimization algorithm	
2	further prevents consecutive load and store operations in the intermediate representation	
3	from being moved into one of the partially live paths.	
1	15. The method of claim 3, further comprising the step of performing a load-	
2	store aliasing optimization.	
1.	16. A computer-readable storage medium having software resident thereon in	
2	the form of computer-readable code executable by a computer to perform the following	
3	steps to translate a plurality of basic blocks of program code:	
4	decoding a plurality of basic blocks of program code:	

intermediate representation to generate an optimized intermediate and generating target code from said optimized intermediate 1 17. The computer-readable storage medium of claim 16, what dead code elimination optimization is performed on basic blocks endire branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, what dead code elimination optimization comprises:	3	generating an intermediate representation for each of said basic blocks of
intermediate representation to generate an optimized intermediate and generating target code from said optimized intermediate 1 17. The computer-readable storage medium of claim 16, what dead code elimination optimization is performed on basic blocks endire branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, what dead code elimination optimization comprises: 3 identifying partially dead register definitions within a beautifying a recursive marking algorithm to mark the chick identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to generate intermediate representation providing an optimized order for ground code. 1 19. The computer-readable storage medium of claim 18, what dead register definition identifying step comprises: 5 for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	6	program code;
generating target code from said optimized intermediate 1 17. The computer-readable storage medium of claim 16, what dead code elimination optimization is performed on basic blocks endired branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, what dead code elimination optimization comprises: 3 identifying partially dead register definitions within a beauty applying a recursive marking algorithm to mark the chick identified partially dead register definitions to produce a set of nodes; and 7 performing code motion optimization algorithm to genesis intermediate representation providing an optimized order for genesis intermediate representation in a basic block, performing liverage and register definition in a basic block containing said register definition in successor basic blocks containing said register definition in successor basic blocks.	7	performing a partial dead code elimination optimization on said
generating target code from said optimized intermediated. 1 17. The computer-readable storage medium of claim 16, who dead code elimination optimization is performed on basic blocks endired branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a beautified partially dead register definitions to produce a set of nodes; and 4 performing code motion optimization algorithm to generate intermediate representation providing an optimized order for generate code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	8	intermediate representation to generate an optimized intermediate representation;
1 17. The computer-readable storage medium of claim 16, who dead code elimination optimization is performed on basic blocks endir branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a brapplying a recursive marking algorithm to mark the christidentified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to generate intermediate representation providing an optimized order for generate code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	9	and
dead code elimination optimization is performed on basic blocks endir branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a bapplying a recursive marking algorithm to mark the chi identified partially dead register definitions to produce a set of nodes; and 7 performing code motion optimization algorithm to gene intermediate representation providing an optimized order for grounder. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing living said register definition in successor basic blocks containing said	10	generating target code from said optimized intermediate representation.
dead code elimination optimization is performed on basic blocks endir branches or computed jumps. 1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a bapplying a recursive marking algorithm to mark the chi identified partially dead register definitions to produce a set of nodes; and 7 performing code motion optimization algorithm to gene intermediate representation providing an optimized order for grounder. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing living said register definition in successor basic blocks containing said		
1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a beautified partially dead register definitions to mark the chi identified partially dead register definitions to produce a set of nodes; and 6 performing code motion optimization algorithm to generate intermediate representation providing an optimized order for generate code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	1	17. The computer-readable storage medium of claim 16, wherein the partial
1 18. The computer-readable storage medium of claim 17, who dead code elimination optimization comprises: 3 identifying partially dead register definitions within a beapplying a recursive marking algorithm to mark the chick identified partially dead register definitions to produce a set of nodes; and 7 performing code motion optimization algorithm to general intermediate representation providing an optimized order for general code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: 3 for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	2	dead code elimination optimization is performed on basic blocks ending in non-computed
dead code elimination optimization comprises: identifying partially dead register definitions within a b applying a recursive marking algorithm to mark the chi identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to gene intermediate representation providing an optimized order for gene code. 1 19. The computer-readable storage medium of claim 18, what dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	3	branches or computed jumps.
dead code elimination optimization comprises: identifying partially dead register definitions within a b applying a recursive marking algorithm to mark the chi identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to gene intermediate representation providing an optimized order for gene code. 1 19. The computer-readable storage medium of claim 18, what dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said		
identifying partially dead register definitions within a bapplying a recursive marking algorithm to mark the chicken identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to general intermediate representation providing an optimized order for general code. 1 19. The computer-readable storage medium of claim 18, which dead register definition identifying step comprises: for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	1	18. The computer-readable storage medium of claim 17, wherein the partial
applying a recursive marking algorithm to mark the chi identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to gene intermediate representation providing an optimized order for gene code. 1 19. The computer-readable storage medium of claim 18, what dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	2	dead code elimination optimization comprises:
identified partially dead register definitions to produce a set of nodes; and performing code motion optimization algorithm to gene intermediate representation providing an optimized order for gene code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	3	identifying partially dead register definitions within a basic block;
performing code motion optimization algorithm to general intermediate representation providing an optimized order for general code. 1 19. The computer-readable storage medium of claim 18, which dead register definition identifying step comprises: for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	4	applying a recursive marking algorithm to mark the child nodes of
performing code motion optimization algorithm to general intermediate representation providing an optimized order for general code. 1 19. The computer-readable storage medium of claim 18, which dead register definition identifying step comprises: for a register definition in a basic block, performing livers and register definition in successor basic blocks containing said	5	identified partially dead register definitions to produce a set of partially dead
intermediate representation providing an optimized order for go code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	6	nodes; and
 code. 1 19. The computer-readable storage medium of claim 18, who dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said 	7	performing code motion optimization algorithm to generate an optimized
1 19. The computer-readable storage medium of claim 18, wl 2 dead register definition identifying step comprises: 3 for a register definition in a basic block, performing live 4 said register definition in successor basic blocks containing said	8	intermediate representation providing an optimized order for generating target
dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	9	code.
dead register definition identifying step comprises: for a register definition in a basic block, performing live said register definition in successor basic blocks containing said		
for a register definition in a basic block, performing live said register definition in successor basic blocks containing said	1	19. The computer-readable storage medium of claim 18, wherein the partially
said register definition in successor basic blocks containing said	2	dead register definition identifying step comprises:
	3	for a register definition in a basic block, performing liveness analysis of
5 branches or computed jumps; and	4	said register definition in successor basic blocks containing said non-computed
• • • •	5	branches or computed jumps; and

Patent Docket No.45256.00052.UTL1

Express Mail No.: EV324254847US

6	identifying said register definition as being partially dead if said registe	r	
7	definition is dead in at least one successor basic block and live in at least one		
8	other successor basic block.		
1	20. The computer-readable storage medium of claim 19, said computer-		
2	readable code further executable for forming a set of identified partially dead register		
3	definitions.		
1	21. The computer-readable storage medium of claim 20, said computer-		
2	readable code further executable for applying a recursive marking algorithm to identify	y	
3	partially dead child nodes in the intermediate representation of identified partially dead	d	
4	register definitions.		
1	22. The computer-readable storage medium of claim 21, wherein said		
2	recursive marking algorithm identifies a node in the intermediate representation as a		
3	partially dead child node by ensuring that the node is not referenced by either a live no	od€	
4	or a live register definition.		
1	23. The computer-readable storage medium of claim 22, wherein said		
2	recursive marking algorithm identifies partially dead child nodes as those nodes which	ì	
3	are only referenced in the intermediate representation by other partially dead nodes or		
4	partially dead register definitions.		
1	24. The computer-readable storage medium of claim 23, wherein said		
2	recursive marking algorithm includes the steps of:		
3	determining a dead count for a child node, wherein the dead count is the	е	
4	number of partially dead nodes referencing the child node in the intermediate		
5	representation;		

6	determining a reference count for the child node, wherein the reference		
7	count is the number of references to the child node in the intermediate		
8	representation; and		
9	identifying a child node as a partially dead when its dead count equals its		
10	reference count.		
1	25. The computer-readable storage medium of claim 21, wherein said		
2	recursive marking algorithm further recursively identifies whether the child nodes of		
3	identified partially dead child nodes are also partially dead.		
1	26. The computer-readable storage medium of claim 18, the code motion		
2	optimization algorithm comprises:		
3	for each identified partially dead register definition,		
4	determining a path of nodes in the intermediate representation for said		
5	partially dead register definition which are live,		
6	discarding the nodes in the intermediate representation for said partially		
7	dead register definition which are dead, and		
8	determining partially live paths of nodes in the intermediate representation		
9	for said partially dead register definition and moving corresponding nodes into		
10	said partially live paths, wherein the nodes in the partially live paths are partially		
11	dead nodes, further wherein a partially live path of nodes exists for each		
12	respective branch or jump.		
1	27. The computer-readable storage medium of claim 26, wherein each node in		
2	the intermediate representation includes an associated variable which identifies with		
3	which partially live path of nodes it is associated		

OC/322219.2

83

1	28.	The computer-readable storage medium of claim 26, wherein said target
2	code generati	ng step comprises:
3		initially generating target code for all fully live nodes of a partially dead
4	registe	er definition; and
5		next generating target code for said partially live paths of nodes in the
6	intern	nediate representation for said partially dead register definition.
		·
1	29.	The computer-readable storage medium of claim 26, wherein the code
2	motion optim	ization algorithm further prevents consecutive load and store operations in
3	the intermedia	ate representation from being moved into one of the partially live paths.
1	30.	The computer-readable storage medium of claim 18, further comprising
2	the step of pe	rforming a load-store aliasing optimization.
1	31.	In combination:
2		a target processor; and
3		translator code for translating a plurality of basic blocks of program code,
4	said tr	ranslator code comprising code executable by said target processor for
5	perfor	ming the following steps:
6		decoding a plurality of basic blocks of program code;
7		generating an intermediate representation for each of said basic
8		blocks of program code;
9		performing a partial dead code elimination optimization on said
10		intermediate representation to generate an optimized intermediate
11		representation; and

12	generating target code from said optimized intermediate
13	representation.
1	32. The combination of claim 31, wherein the partial dead code elimination
2	optimization is performed on basic blocks ending in non-computed branches or compute
3	jumps.
1	33. The combination of claim 32, wherein the partial dead code elimination
2	optimization comprises:
3	identifying partially dead register definitions within a basic block;
4	applying a recursive marking algorithm to mark the child nodes of
5	identified partially dead register definitions to produce a set of partially dead
6	nodes; and
7	performing code motion optimization algorithm to generate an optimized
8	intermediate representation providing an optimized order for generating target
9	code.
1	34. The combination of claim 33, wherein the partially dead register definition
2	identifying step comprises:
3	for a register definition in a basic block, performing liveness analysis of
4	said register definition in successor basic blocks containing said non-computed
5	branches or computed jumps; and
6	identifying said register definition as being partially dead if said register
7	definition is dead in at least one successor basic block and live in at least one
8	other successor basic block.

Docket No.45256.00052.UTL1

Express Mail No.: EV324254847US

1	35.	The combination of claim 34, said translator code further comprising code
2	executable by	y said target processor for forming a set of identified partially dead register
3	definitions.	
1	36.	The combination of claim 35, said translator code further comprising code
2	executable by	y said target processor for applying a recursive marking algorithm to identify
3	partially dead	child nodes in the intermediate representation of identified partially dead
4	register defin	itions.
1	37.	The combination of claim 36, wherein said recursive marking algorithm
2	identifies a n	ode in the intermediate representation as a partially dead child node by
3	ensuring that	the node is not referenced by either a live node or a live register definition.
1	38.	The combination of claim 36, wherein said recursive marking algorithm
2	identifies par	tially dead child nodes as those nodes which are only referenced in the
3	intermediate	representation by other partially dead nodes or partially dead register
4	definitions.	
1	39.	The combination of claim 38, wherein said recursive marking algorithm
2	includes the	steps of:
3		determining a dead count for a child node, wherein the dead count is the
4	numb	er of partially dead nodes referencing the child node in the intermediate
5	repres	sentation;
6	÷	determining a reference count for the child node, wherein the reference
7	count	is the number of references to the child node in the intermediate
8	renres	sentation: and

9		identifying a child node as a partially dead when its dead count equals its
10	refere	ence count.
1	40.	The combination of claim 3, wherein said recursive marking algorithm
2	further recurs	sively identifies whether the child nodes of identified partially dead child
3	nodes are als	o partially dead.
1	41.	The combination of claim 33, the code motion optimization algorithm
2	comprises:	
3		for each identified partially dead register definition,
4		determining a path of nodes in the intermediate representation for said
5	partia	lly dead register definition which are live,
6		discarding the nodes in the intermediate representation for said partially
7	dead register definition which are dead, and	
8		determining partially live paths of nodes in the intermediate representation
9	for sa	id partially dead register definition and moving corresponding nodes into
10	said p	partially live paths, wherein the nodes in the partially live paths are partially
11	dead	nodes, further wherein a partially live path of nodes exists for each
12	respec	ctive branch or jump.
1	42.	The combination of claim 41, wherein each node in the intermediate
2	representation	n includes an associated variable which identifies with which partially live
3	path of nodes	s it is associated.
1	43.	The combination of claim 41, wherein said target code generating step
2	comprises:	
3		initially generating target code for all fully live nodes of a partially dead
4	regist	er definition; and

Docket No.45256.00052.UTL1

Express Mail No.: EV324254847US

next generating target code for said partially live paths of nodes in the 5 6 intermediate representation for said partially dead register definition.

- The combination of claim 41, wherein the code motion optimization 1 44. 2 algorithm further prevents consecutive load and store operations in the intermediate representation from being moved into one of the partially live paths. 3
- 1 45. The combination of claim 33, said translator code further comprising code 2 executable by said target processor for performing a load-store aliasing optimization.